17th Computational Fluid Dynamics Conference, June 6–9, 2005, Toronto, ON

Adjoint Algorithm for CAD-Based Shape Optimization Using a Cartesian Method

Marian Nemec[*]
ELORET Corp., Moffett Field, CA 94035

Michael J. Aftosmis[†]
NASA Ames Research Center, Moffett Field, CA 94035

We focus on the computation of objective function gradients using a discrete adjoint method for embedded-boundary Cartesian meshes. This is a significant step in our research toward using computer-aided design (CAD) directly for adjoint-based gradient computations. The approach treats the Cartesian mesh as a rigid structure. Under this assumption, accurate evaluation of mesh sensitivities depends directly on the formulation within the layer of non-uniform cells, or cut-cells, immediately adjacent to the surface. The formulation is based on the linearization of a simple geometric constructor, which decouples the computation of shape sensitivities of the surface triangulation from the cut-cell sensitivities. As a result, the method is well suited to CAD-based optimization using parametric solid models. Detailed verification studies of gradient accuracy are presented for several two- and three-dimensional shape optimization problems.

I. Introduction

Automation and turn-around time are critical factors in the use of Euler and Navier–Stokes simulation tools for design optimization. When a designer successfully reduces a real-life design problem to a well-posed optimization problem, i.e., asks the computer a sufficiently simple question, the optimization procedure should rapidly identify design improvements that complement the intuition and experience of the designer. This requires a robust optimization procedure, which means that distractions such as failures in geometry manipulation, mesh generation, sensitivity computation, and flow-solver convergence are eliminated for any admissible shape changes in geometry.

Cartesian methods are perhaps the most promising approach for addressing the issues of flow solution automation.[1–5] In these methods, the discretization of the wetted surface is decoupled from that of the volume mesh. This not only enables fast and robust mesh generation for geometry of arbitrary complexity,[6,7] but also facilitates access to geometry modeling and manipulation using parametric computer-aided design (CAD). The degree of coupling between the CAD system and the optimization procedure, as well as the related issue of geometry parameterization, are topics of current research.[8–13] In previous work,[14] we have demonstrated the effectiveness of a direct CAD interface[15,16] in conjunction with an embedded-boundary Cartesian method[17,18] for optimization problems involving large shape modifications, including changes in topology.

Our present goal is to combine the automation capabilities of Cartesian methods with an efficient computation of design sensitivities, thereby reducing the turn-around time of the optimization process. We address this issue using the adjoint method,[19,20] where the computational cost of the design sensitivities, or objective function gradients, is essentially independent of the number of design variables. In contrast to implementations on body-fitted structured and unstructured meshes,[21–29] adjoint implementations on embedded-boundary Cartesian meshes require additional attention at the intersection of the volume mesh with the geometry. Examples include the work of Melvin *et al.*,[30] who developed an adjoint formulation for

[*] Research Scientist, Applications Branch, Advanced Supercomputing Division, MS T27B; mnemec@mail.arc.nasa.gov. Member AIAA.

[†] Research Scientist, maftosmis@mail.arc.nasa.gov. Senior Member AIAA.

the Cartesian TRANAIR code,[1] and Dadone and Grossman,[31,32] who presented an adjoint formulation for the Euler equations using a ghost-cell method to enforce the wall boundary conditions.

In Ref. 33, we presented an accurate and efficient algorithm for the solution of the adjoint Euler equations discretized on Cartesian meshes with embedded, cut-cell boundaries. The accuracy of the gradient computation was verified using several three-dimensional test cases at transonic and supersonic flow conditions. These preliminary investigations restricted their attention to mostly global design variables, such as the angle of incidence, where the shape of the geometry remained constant. In this work, we extend our adjoint formulation to include general shape changes. While this is a straightforward extension in body-fitted approaches, Cartesian shape optimization demands the linearization of the cut-cell boundary. The analysis section presents a novel approach to this issue, where we introduce an approximate linearization of the coupling between the surface triangulation and the Cartesian cells. We also discuss the integration of the adjoint algorithm with a direct CAD interface of Haimes *et al.*[15,16] An important aspect of the CAD interface is the computation of surface shape sensitivities for triangulations based on parametric solid models. We present initial results of the accuracy of the gradient computation for several two- and three-dimensional test cases.

II. Optimization Problem

The aerodynamic optimization problem we consider in this work consists of determining values of design variables that minimize a given objective function

$$\min_{X} \ \mathcal{J}(X, Q) \tag{1}$$

where $\mathcal{J}$ represents a scalar objective function defined by a surface integral, for example lift or drag, X denotes a scalar design variable, for example a shape parameter of the wetted surface, and $Q = [\rho, \rho u, \rho v, \rho w, \rho E]^{\mathrm{T}}$ denotes the continuous flow variables. The flow variables are forced to satisfy the steady-state three-dimensional Euler equations within a feasible region of the design space Ω

$$\mathcal{F}(X, Q) = 0 \qquad \forall \ X \in \Omega \tag{2}$$

which implicitly defines $Q = f(X)$. Our goal is to compute a reliable approximation to the objective function gradient $\mathrm{d}\mathcal{J}/\mathrm{d}X$. We proceed by using the discrete adjoint method, where the governing equations, Eqs. 1 and 2, are first discretized and then differentiated.

A. Discrete Flow Equations and Solution Method

The discretization of the Euler equations uses a second-order accurate finite-volume method on a multilevel Cartesian mesh with embedded boundaries. The mesh consists of regular hexahedral cells, except for a layer of body-intersecting cells, or cut-cells, that are arbitrary polyhedra adjacent to the boundaries. A cell-centered approach is used, where the control volumes correspond to the mesh cells and the cell-averaged value of Q, denoted by $\bar{Q}$, is located at the centroid of each cell. The flux residual in each cell i is expressed as

$$R_i = \sum_{\mathrm{j} \in \mathrm{V_i}} \mathbf{F}_j \cdot \hat{\mathbf{n}}_j A_j \tag{3}$$

where j denotes the jth face of volume V_i with area A and unit normal $\hat{\mathbf{n}}$, and $\mathbf{F}$ represents the numerical flux function. The flux function is evaluated at the face centroids using the flux-vector splitting approach of van Leer.[34] The boundary conditions are enforced weakly by appropriate modifications of the reconstructed state and boundary flux. The resulting discrete system of equations is given by

$$\vec{R}(\vec{Q}, \vec{M}, X) = 0 \tag{4}$$

where $\vec{Q} = [\bar{Q}_1, \bar{Q}_2, \ldots, \bar{Q}_N]^{\mathrm{T}}$ is the discrete solution vector for all N cells of a given mesh $\vec{M}$, and $\vec{R}$ is the flux residual vector.

Steady-state flow solutions are obtained using a five-stage Runge–Kutta scheme with local time stepping, multigrid, and a highly-scalable domain decomposition scheme for parallel computing. For further details on the spatial discretization and flow solution, see Aftosmis *et al.*[7,17,18] and Berger *et al.*[35,36]

Note that design variables appear directly in Eq. 4 only when they involve parameters that do not change the computational domain, such as the Mach number, angle of incidence, and side-slip angle. The influence of shape design variables on the residuals is implicit via the computational mesh

$$\vec{M} = f[\vec{T}(X)] \tag{5}$$

where $\vec{T}$ denotes a triangulation for the surface model. The functional dependence of the triangulation on the design variables is determined by the geometry parameterization scheme and is explained further in Sec. III.

B. Discrete Adjoint Method

Combining Eqs. 4 and 5 and differentiating about a steady-state solution $\vec{Q}$ gives the gradient of the discrete objective function $\mathcal{J}(X, \vec{M}, \vec{Q})$

$$\frac{\mathrm{d}\mathcal{J}}{\mathrm{d}X} = \frac{\partial \mathcal{J}}{\partial X} + \underbrace{\frac{\partial \mathcal{J}}{\partial \vec{M}} \frac{\partial \vec{M}}{\partial \vec{T}} \frac{\partial \vec{T}}{\partial X}}_{\mathrm{A}} - \vec{\psi}^{\mathrm{T}} \left(\frac{\partial \vec{R}}{\partial X} + \underbrace{\frac{\partial \vec{R}}{\partial \vec{M}} \frac{\partial \vec{M}}{\partial \vec{T}} \frac{\partial \vec{T}}{\partial X}}_{\mathrm{B}} \right) \tag{6}$$

where the vector $\vec{\psi}$ represents adjoint variables given by the adjoint equation

$$\frac{\partial \vec{R}}{\partial \vec{Q}}^{\mathrm{T}} \vec{\psi} = \frac{\partial \mathcal{J}}{\partial \vec{Q}}^{\mathrm{T}} \tag{7}$$

We emphasize that the adjoint equation is independent of the design variables, i.e., the partial derivatives with respect to the flow variables are evaluated at constant X. Hence, for problems with many design variables and one objective function, the use of the adjoint method is attractive since only one solution of the large linear system of Eq. 7 is required.

The solution algorithm for the adjoint equation leverages the Runge-Kutta time-marching scheme and the parallel multigrid method of the flow solver. The algorithm is implemented using the duality-preserving approach,[37] such that the asymptotic convergence rate of the discrete adjoint is identical to that of the flow solver. The matrix-vector products associated with the flow-Jacobian matrix, left side of Eq. 7, are computed on-the-fly using a two-pass strategy over the faces of the mesh. The flow-Jacobian matrix, as well as the term $\partial \mathcal{J}/\partial \vec{Q}$ in Eq. 7, are derived by hand, where we neglect the linearization of the limiter function used in the solution reconstruction procedure. Overall, the CPU time per iteration and memory usage of the adjoint solver are roughly equivalent to the flow solver. For further details on the adjoint formulation and solution procedure see Ref. 33.

The computation of partial derivative terms $\partial \mathcal{J}/\partial X$ and $\partial \vec{R}/\partial X$ in Eq. 6 is straightforward, since these terms do not involve derivatives of the surface shape. For example, the derivation of $\partial \mathcal{J}/\partial X$ for the angle of incidence design variable involves the differentiation of the transformation matrix between the body frame (normal, lateral, and axial forces) and aerodynamic frame (lift, side, and drag forces) for objective functions based on the aerodynamic coefficients. The corresponding residual sensitivity term is given by

$$\frac{\partial \vec{R}}{\partial X} = \frac{\partial \vec{R}}{\partial \vec{U}_{\mathrm{bc}}} \frac{\partial \vec{U}_{\mathrm{bc}}}{\partial \vec{U}_{\infty}} \frac{\partial \vec{U}_{\infty}}{\partial \alpha} \tag{8}$$

where the first two right-hand side terms represent the Jacobian of the flux function (see Eq. 3) and the Jacobian of the primitive variables associated with the far-field Riemann invariants, respectively. These two terms are reused directly from the adjoint formulation. The last term is the linearization of the freestream state vector, which involves simple trigonometric relations.

The remaining partial derivative terms in Eq. 6, labeled as A and B, represent the differentiation of the objective function and residual equations with respect to design variables that alter the surface shape. An accurate computation of these terms is presented in the following section.

III. Objective and Residual Shape Sensitivities

The differentiation of the objective function and residual equations with respect to shape design variables involves the interaction of the surface deformations with the triangulation and the volume mesh. An advantage of our Cartesian approach is that the computation of the surface shape sensitivities, i.e., the term $\partial\vec{T}/\partial X$ in Eq. 6, is cleanly isolated from the residual equations. This is because there is no prescribed connectivity between the volume mesh and the triangulation, which is particularly well-suited for CAD-based geometry control. In subsection A, we outline our approach to geometry manipulation and surface triangulation using a direct CAD interface, and then explain the computation of surface shape sensitivities. In subsection B, we address the binding of the surface to the volume mesh to obtain sensitivities in the cut-cell layer.

A. Shape Sensitivity of Surface Triangulation

Our primary method for surface modeling, control, and triangulation is based on the Computational Analysis and PRogramming Interface (CAPRI) developed by Haimes *et al.*[15,16] CAPRI provides a unified and direct access to most parametric CAD systems. The ability to control CAD solid models is accomplished by exposing the master-model feature tree of parametric parts and assemblies. Design variables can be associated directly with the exposed parameters, or indirectly with splines, such as airfoil sections that are lofted to define a wing.

Upon a successful regeneration of the CAD-model, CAPRI provides an associated water-tight triangulation for each component. The Cartesian position of each vertex in the triangulation, for each face of the CAD model, is given by its parametric values

$$(x, y, z) = f(u, v) \tag{9}$$

The triangulation algorithm is robust, and generates both isotropic and anisotropic triangulations depending on the topology of the CAD faces.[38,39]

Since analytic differentiation of the geometry constructors within the CAD system is presently not possible, we rely on a centered-difference approximation to determine the derivative of each vertex of the triangulation with respect to the design variables. The first step of this procedure involves the regeneration and triangulation of the CAD model to reflect the current values of the design variables, thereby obtaining a baseline model. For each CAD face, we normalize the (u, v) vertex values such that $u, v \in [0, 1]$. The next step involves two additional model regenerations and triangulations for each design variable, which correspond to the plus and minus perturbations. To form the finite-difference approximation, we interrogate the perturbed models using the normalized (u, v) values re-scaled by their new (u, v) range. Hence, we obtain a one-to-one correspondence between vertex coordinates in the physical space for the baseline model and its perturbations. A tacit assumption in this procedure is that the face topology of the baseline and perturbed geometries remains the same. We check this condition after each regeneration of the model and attempt to reduce the stepsize or use one-sided differences if necessary.

The finite-difference procedure is efficient since we query the triangulation in the parameter space, i.e., we only require forward solutions of Eq. 9. An alternate approach is the use of "snap" operations in the physical space to locate nearest vertex coordinates on the baseline and perturbed surfaces. This procedure is less efficient since it involves an inverse solution of Eq. 9. Furthermore, due to internal tolerances of the CAD system, snap operations may introduce excessive noise into the evaluation of the derivative.

In addition to the CAPRI interface, we also use an analytic geometry parameterization scheme for verification testing. The scheme is limited to airfoil design problems, where we modify the thickness distribution of airfoils using a *sine* bump function[40]

$$\Delta V_y = X \sin\left(\pi \frac{V_x - V_x^{\mathrm{s}}}{V_x^{\mathrm{e}} - V_x^{\mathrm{s}}}\right) \tag{10}$$

where ΔV_y is the change in the y-component of vertex V, and V_x^{s} and V_x^{e} denote the starting and ending chord positions of the bump function, respectively. The shape sensitivity is given by

$$\frac{\partial V_y}{\partial X} = \sin\left(\pi \frac{V_x - V_x^{\mathrm{s}}}{V_x^{\mathrm{e}} - V_x^{\mathrm{s}}}\right) \tag{11}$$

B. Sensitivity of Cut-Cells to Shape Deformations

Before presenting the computation of objective and residual shape sensitivities on Cartesian meshes, it is instructive to briefly review the implementations on body-fitted structured and unstructured meshes. These methods typically employ mesh-perturbation strategies that smoothly deform the volume mesh in response to changes in the surface geometry, as sketched in Fig. 1. The linearization of a mesh-perturbation scheme is represented by the term $\partial\vec{M}/\partial\vec{T}$ in Eq. 6.

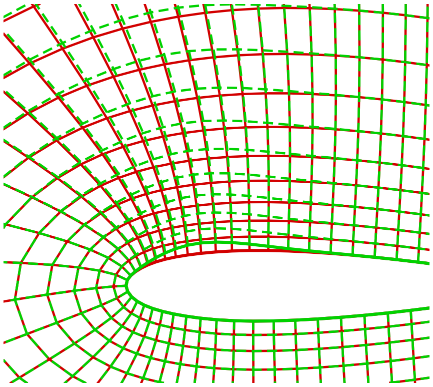


Figure 1. Mesh-perturbation approach (Baseline airfoil and grid are solid red, perturbation is dashed green)

Detailed studies have been performed by Anderson and Venkatakrishnan[21] and later by Nielsen and Anderson[41] to determine the effect of mesh sensitivities on accurate gradient computations. In the discrete formulation of the gradient, as in Eq. 6, the mesh perturbations could be limited to just the surface boundary cells, or extended gradually to the far-field. In the continuous formulation, there are no mesh sensitivities since all terms are evaluated on the surface. The aforementioned studies show that for local shape deformations, away from geometric singularities, the extent of mesh sensitivities can be limited to just the boundary cells. However, for more general problems, such as cases that involve rigid body motion, the use of a mesh-perturbation strategy is important. In related work, Giles *et al.*[26] presented a number of examples that emphasize the fact that the grid convergence of an objective function does not guarantee the grid convergence of its gradient. Jameson and Kim[42] introduced a reduced adjoint gradient formulation where only surface perturbations are considered. Their results indicate that for shape optimization problems, the reduced approach works well. Similar results are also presented by Soto and Löhner.[43]

Turning our attention back to embedded-boundary Cartesian methods, it seems natural to compute objective and residual shape sensitivities only for the boundary cells. For example, Huffman *et al.*[44] developed and linearized a transpiration boundary condition for the TRANAIR code[30, 44] to approximate the effects of the changing boundary surface. They obtained good results for problems where the design variables involve shape changes normal to the surface, including wing-twist distributions. Dadone and Grossman[31] use finite differences in conjunction with their Cartesian ghost-cell method to approximate shape sensitivities in the boundary cells.

In the present Cartesian method, the use of finite-differences is complicated by the presence of cut-cells that arise from the arbitrary intersection of the surface with the Cartesian cells. The appearance and disappearance of cut-cells is likely for any finite stepsize. We propose an approach that circumvents these difficulties and is significantly more efficient. The approach is based on the linearization of a geometry constructor that closely approximates the mesh generation process. We note at the outset that our main focus is on the cut-cell region, and therefore we leave the subject of mesh-perturbation schemes for future work.

The cut-cell generation procedure involves computing the intersection of the surface triangulation with the faces of Cartesian hexahedra. Details of this procedure are described in Ref. 6. We approximate the cut-cell geometry using the intersection of surface triangles with only the Cartesian edges and neglect contributions from the faces. This is illustrated in Fig. 2, where a Cartesian edge $\overline{\mathbf{de}}$ of a hexahedron intersects triangle $(\mathbf{a},\mathbf{b},\mathbf{c})$, creating in a pierce point $\mathbf{P}$. We use bold type to represent Cartesian position vectors.

The linearization of the pierce point location with respect to a shape perturbation could be obtained directly from the CAD system using the approach described in the previous section. This would involve the use of "snap" operations to locate the pierce point in the (u, v) space of the appropriate CAD face. Alternatively, the pierce point location can be defined using a set of parametric equations for the triangle $(\mathbf{a},\mathbf{b},\mathbf{c})$ and the line segment $\overline{\mathbf{de}}$, which is the approach we pursue in this work. The location of $\mathbf{P}$ along the segment $\overline{\mathbf{de}}$ is given by

$$\mathbf{P} = \mathbf{d} + s^* \cdot \mathbf{D} \tag{12}$$

where $\mathbf{D} = \mathbf{e} - \mathbf{d}$ and $s^* = f(\mathbf{a},\mathbf{b},\mathbf{c},\mathbf{d},\mathbf{e})$ represents the parametric value of the intersection point. The expression for s^* is derived in Ref. 6 and reproduced in the Appendix. The linearization of the pierce point is given by

$$\frac{\partial \mathbf{P}}{\partial X} = \frac{\partial s^*}{\partial X}\mathbf{D} \tag{13}$$

Note that only one component of $\mathbf{D}$ is non-zero, since the Cartesian mesh points are coordinate aligned. Furthermore, the points $\mathbf{d}$ and $\mathbf{e}$ are independent of X, i.e., the Cartesian mesh is rigid, and consequently the linearization of s^* is given by

$$\frac{\partial s^*}{\partial X} = \frac{\partial s^*}{\partial \mathbf{a}}\frac{\partial \mathbf{a}}{\partial X} + \frac{\partial s^*}{\partial \mathbf{b}}\frac{\partial \mathbf{b}}{\partial X} + \frac{\partial s^*}{\partial \mathbf{c}}\frac{\partial \mathbf{c}}{\partial X} \tag{14}$$

This linearization represents the change in the location of $\mathbf{P}$ along the edge $\overline{\mathbf{de}}$ exclusively as a function of shape sensitivities at the vertices $(\mathbf{a}, \mathbf{b}, \mathbf{c})$.

The linearization of the pierce points is used to determine all the required geometric sensitivities in the linearization of the residual equations. Referring to Eq. 3, first-order spatial discretization requires the derivative of the face areas and normals. For second-order discretization, we require additional linearizations for the location of face centroids, volume centroids, and the reconstruction distances used in the evaluation of flow-solution gradients. As a result, a perturbation of the surface shape influences the residuals not only in the cut-cells, but also in the first and second nearest neighbouring cells.

The computation of the surface normal is based on a planar approximation to the variation of the triangulation within each cut-cell. This agglomerated normal vector is computed using the divergence theorem, which requires the geometric closure of each cut-cell

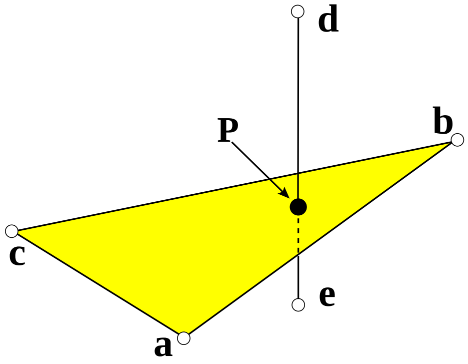


Figure 2. Intersection of a triangle with a Cartesian edge defining a pierce point P

$$\sum_{\mathrm{j}\in \mathrm{V_i}} \mathbf{n}_j = \sum_{\mathrm{j}\in \mathrm{V_i}} A_j \cdot \hat{\mathbf{n}}_j = 0 \tag{15}$$

where $\mathbf{n}$ denotes an area scaled normal vector. The sum is performed over the Cartesian faces of the cell, which must equal the agglomerated normal vector. Since geometric closure of a cut-cell should be satisfied for any shape perturbation, we linearize Eq. 15 to obtain

$$\sum_{\mathrm{j}\in \mathrm{V_i}} \frac{\partial}{\partial X}\left(A_j \cdot \hat{\mathbf{n}}_j\right) = 0 \tag{16}$$

The derivative of the Cartesian face areas is computed based on the linearization of the pierce points. The pierce points, in conjunction with the corner points of the cut-cell, form polygons on each Cartesian face. The areas of the polygons are computed by subdividing each polygon into triangles. Central to this procedure is the linearization of an area of a triangle given by the cross-product of two vectors. The gradient computations presented in the following section are based on a first-order linearization of geometry dependent terms in the residual equations. We leave the extension to second-order discretization for future work.

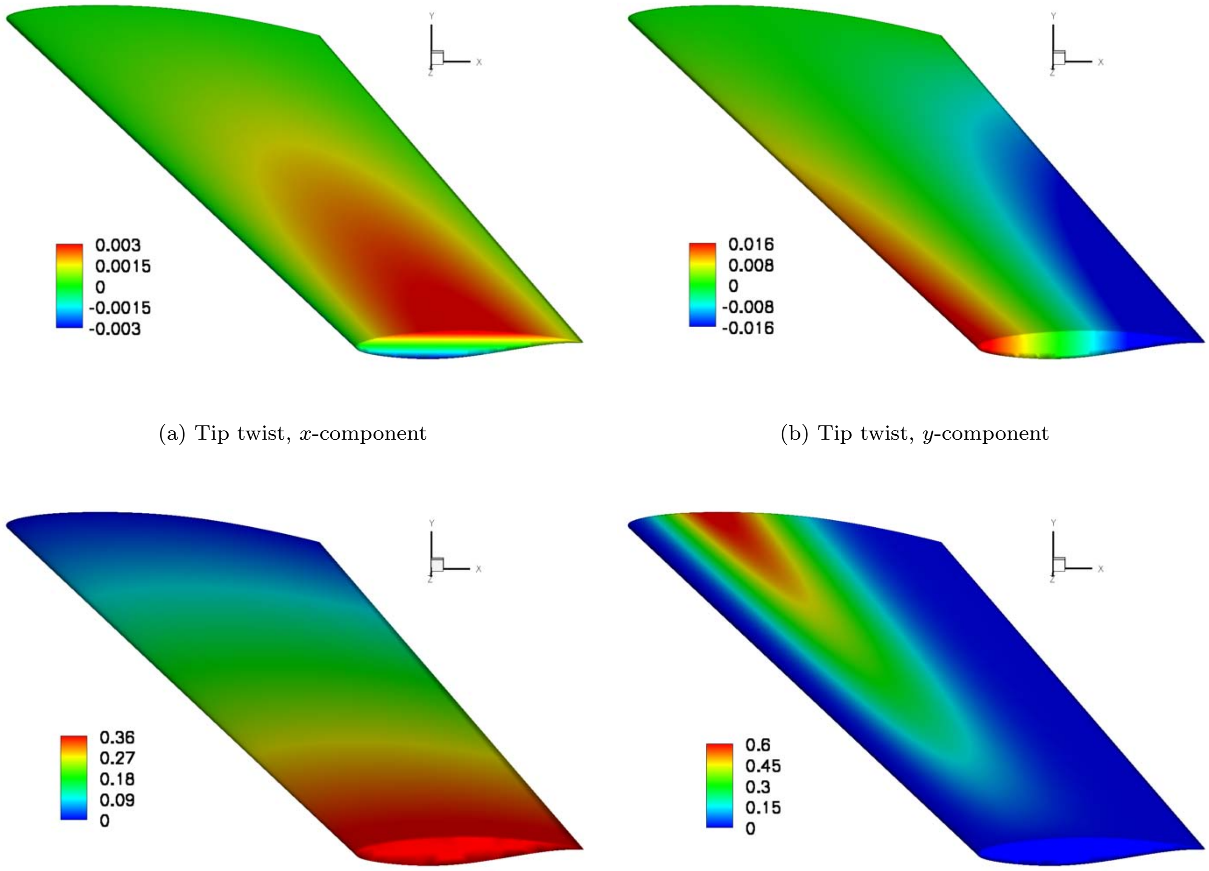
IV. Results and Discussion

We present several two- and three-dimensional test cases to verify the accuracy of the cut-cell linearization and gradient computations. The Pro/ENGINEER® Wildfire CAD system is used to create all geometry models.

A. Surface Shape Sensitivities

Figure 3 shows the sensitivity of the surface triangulation to planform and shape design variables, $\partial \vec{T}/\partial X$ in Eq. 6, for a generic wing model. The sensitivities are obtained using CAPRI as described in Sec. IIIA. The relative stepsize used for the centered-difference sensitivity computations is given by $\epsilon = \eta^{1/3}$, where η denotes single-precision machine accuracy. We also tested a stepsize based on the solid-model accuracy obtained from the Pro/ENGINEER kernel. The sensitivities remained essentially unchanged.

The design variable for the local shape deformation shown in Fig. 3(d) is the y-component of a B-spline control point of the root airfoil section. Note that for this design variable, we also obtained sensitivities in the z-direction (not shown). This appears to be related to spline normalizations performed by the CAD system and is presently under investigation. In terms of performance, the CAD-model regeneration requires approximately one second, while 15 seconds[a] are required for surface triangulation (containing roughly 64,800 triangles).



(a) Tip twist, x-component

(b) Tip twist, y-component

(c) Sweep, x-component

(d) Root shape, y-component

Figure 3. Triangulation shape sensitivities, $\frac{\partial \vec{T}}{\partial X}$, for various design variables

B. Sensitivity of Surface Normals

The derivatives of the agglomerated surface normal using Eq. 15 are shown in Fig. 4 for the NACA 0010 airfoil. The design variable is a shape change as defined by Eq. 10 over the upper surface of the airfoil. Consequently, the derivatives are small and change direction near the mid chord of the airfoil, and reach large values near the leading and trailing edges. Note the correct treatment of split cells, i.e., cells with multiple flow regions, near the trailing edge of the airfoil.

C. NACA 0012 Airfoil

This verification test case involves a lift-enhancement design problem for the NACA 0012 airfoil at subsonic conditions. Our goal is to verify the accuracy of the objective function gradient computation. The freestream

[a] Intel® Xeon™ 2 GHz processor, LINUX OS

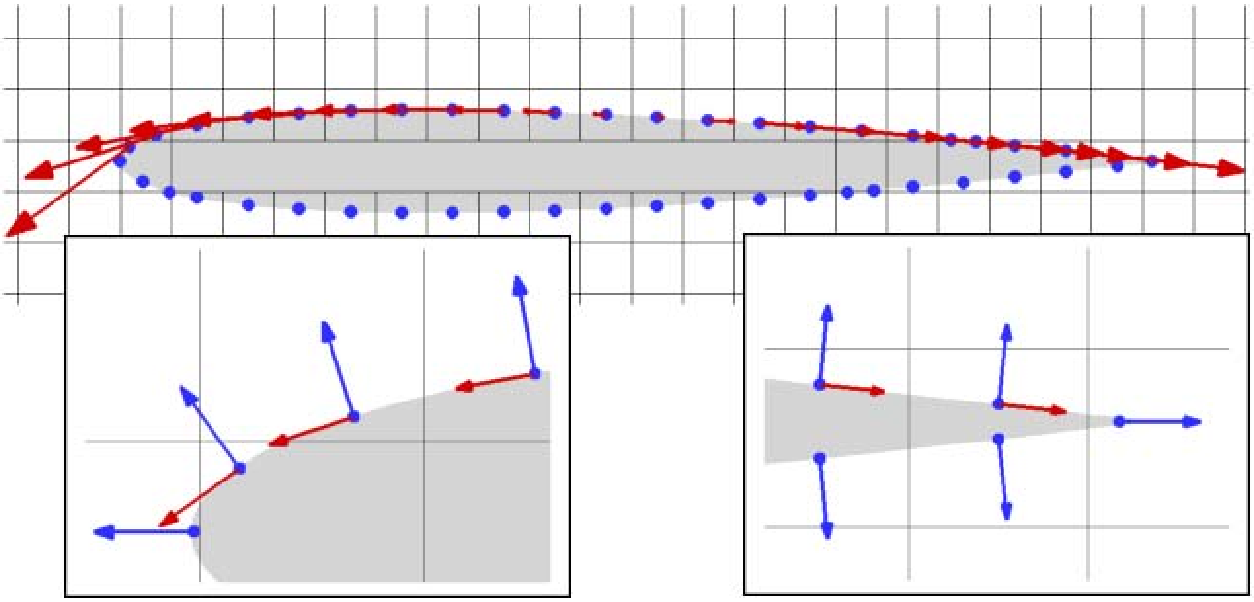


Figure 4. Derivatives of agglomerated surface unit normal (red vectors) with respect to an upper-surface shape design variable. Also shown are agglomerated cut-cell centroids (blue symbols) and agglomerated unit normals (blue vectors)

conditions are $M_\infty = 0.6$ and $\alpha = 2°$. The two-dimensional mesh contains roughly $9,000$ cells after 12 levels of refinement. The objective function is given by

$$\mathcal{J} = \left(1 - \frac{C_\mathrm{L}}{C_\mathrm{L}^*}\right)^2 \tag{17}$$

where $C_\mathrm{L}^* = 0.5$ denotes the target lift coefficient. We consider a single shape design variable as defined by Eq. 10, where the extent of the bump function is from 3%c to 60%c on the upper surface of the airfoil.

Table 1 shows the accuracy of the computed gradients. We perform two numerical experiments using first- and second-order spatial discretization. We use centered-differences to verify the adjoint gradient and present two stepsize values. The agreement between the finite-difference and adjoint gradients is excellent for first-order spatial discretization. Note that the agreement improves as the stepsize is reduced, which is primarily due to a more consistent volume-mesh topology between the baseline and perturbed geometries. The agreement of the gradient values is not as good for the second-order discretization. In this numerical experiment, the flow and adjoint solutions are both based on second-order spatial discretization (limiter is not used); however, we have yet to linearize the second-order geometry-dependent terms in the evaluation of the objective function and residual equations. Nevertheless, the present linearization appears to provide a reasonable estimate of the gradient for the second-order accurate scheme.

Table 1. Gradient accuracy for the NACA 0012 airfoil

Spatial	Adjoint	Finite-Difference	Finite-Difference
Discretization		$\epsilon = 0.001$	$\epsilon = 0.0001$
$\mathcal{O}(h)$	-3.94422	-3.91982	-3.94422
$\mathcal{O}(h^2)$	-3.67299	-3.52984	-3.3363

D. NACA 0012 Wing

The final test case for gradient accuracy verification involves an isolated wing at transonic flow conditions $M_\infty = 0.84$ and $\alpha = 3.06°$. The wing geometry is constructed from a generic CAD model with linearly lofted NACA 0012 root and tip sections, a taper ratio of 0.7, aspect ratio of 11, sweep of 15°, and root and tip-twist angles of 3° and −1.5°, respectively. The volume mesh contains 499,716 cells and uses a symmetry plane. Convergence to steady-state is achieved using 64 processors[b], a 4-level W-cycle multigrid with one pre- and one post-smoothing pass, and a CFL number of 1.2. Partial updates of the flow gradients are also used, i.e., the flow gradients are updated only on the first stage of the RK5 scheme. The second-order accurate spatial discretization uses the van Leer limiter. The Mach number contours of the flow solution are shown in Fig. 5.

The objective function represents a lift-constrained drag minimization problem

$$\mathcal{J} = \left(1 - \frac{C_\mathrm{L}}{C_\mathrm{L}^*}\right)^2 + 0.01\left(1 - \frac{C_\mathrm{D}}{C_\mathrm{D}^*}\right)^2 \tag{18}$$

where $C_\mathrm{L}^* = 0.327$ denotes the initial lift coefficient and $C_\mathrm{D}^* = 0.01$. The initial drag coefficient is 0.0487. The design variables are the angle of incidence and the tip-twist angle of the wing. The surface shape sensitivities are generated using CAPRI and are similar to the results shown in Figs. 3(a) and 3(b).

Convergence of the flow and adjoint equations are shown in Fig. 6. Approximately the first 50 multigrid cycles of the flow solution and the first 75 cycles of the adjoint solution correspond to full-multigrid startup. As shown in Fig. 6(b), the startup time is almost negligible. Furthermore, the convergence characteristics of both solvers are quite similar, with a reduction in residual of over six orders of magnitude in 60 seconds. Note that the flow solver takes advantage of specific compiler directives for code optimization. These directives are not yet implemented in the adjoint code.

Objective function gradient accuracy is presented in Table 2 for both design variables. The agreement between the adjoint and centered-difference gradient values is good. The small differences are attributed to the constant limiter assumption in the linearization of the discrete flow equations, as well as the dependence of the finite-difference gradient on the stepsize due to non-smooth changes in the shock location. In addition, recall that for the tip-twist design variable, the linearization of the discrete flow equations with respect to geometry parameters is presently limited to first-order terms. The interpretation of the descent direction, which is the negative of the gradient vector, reflects the fact that the objective function is dominated by the drag term. Hence, a reduction in the angle of attack and an increase in the washout of the wing should lower the shock strength and therefore reduce the drag penalty.

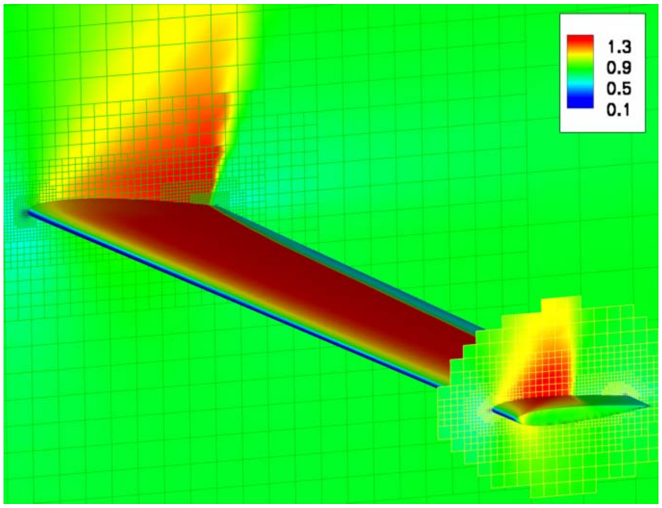
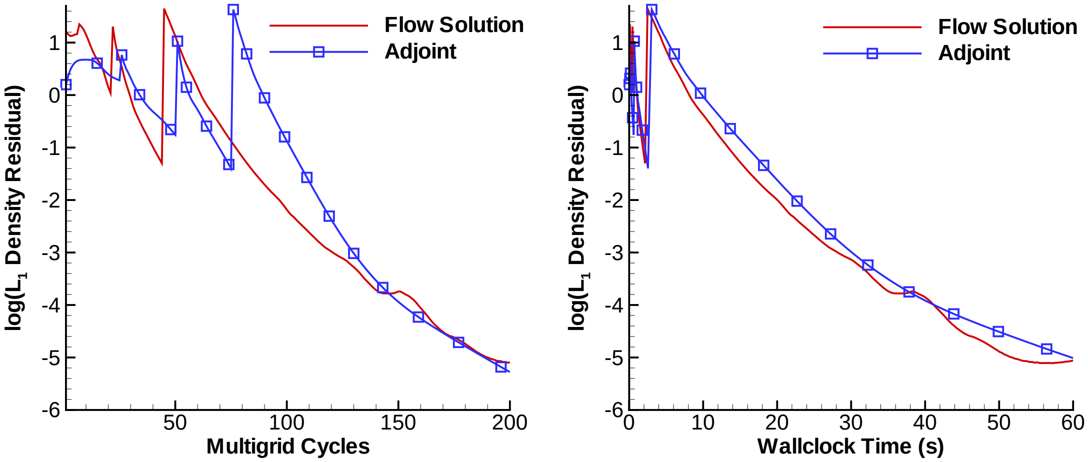


Figure 5. Mach contours for the NACA 0012 Wing ($M_\infty = 0.84$, $\alpha = 3.06°$)

[b] Intel® 1.5 GHz IA-64 Itanium 2 processor



(a) Multigrid Cycles (4-level multigrid with full-multigrid startup)

(b) Wallclock Time (64 Intel® 1.5 GHz Itanium 2 processors, mesh size 499,716 cells)

Figure 6. Convergence histories for the NACA 0012 Wing ($M_\infty = 0.84$, $\alpha = 3.06°$)

Table 2. Gradient accuracy for the NACA 0012 Wing

Design Variable	Adjoint	Finite-Difference
α	0.172276	0.177888
Tip Twist	0.062750	0.063810

V. Summary and Future Work

We have presented a discrete adjoint algorithm for computing aerodynamic shape sensitivities of problems governed by the three-dimensional Euler equations. The equations were discretized on an embedded-boundary Cartesian mesh. We focused on the linearization of the discrete residual equations with respect to shape design variables. A simple geometric constructor was introduced to approximate the binding of the triangulation to the cut-cells of the Cartesian mesh. The shape sensitivities of the triangulation were obtained using a direct-CAD interface. We verified the accuracy of the gradient computation by comparison with centered-difference approximations and obtained good agreement. In the final verification test case, which was based on a wing geometry, the combined flow and adjoint solution time was only two minutes. This is an encouraging result, since it indicates a fast turn-around time for practical design problems with many design variables.

Future work focuses on extending the shape linearization to include second-order geometry terms of the residual equations and investigating the use of mesh-perturbation schemes. In addition, the computation of surface shape sensitivities for configurations with multiple intersecting components is an important issue in our CAD-based approach.

VI. Acknowledgments

The authors gratefully acknowledge Robert Haimes (MIT) for his assistance with CAPRI. The authors would also like to thank Marsha Berger (NYU), Jonathan Goodman (NYU), and Scott Murman (ELORET) for helpful discussions. This work was supported by the NASA Ames Research Center contract NAS2-00062.

Appendix: Intersection of Cartesian Edge with Triangle

The parametric equations for the construction of pierce points are presented in Ref. 6. Referring to Fig. 2, the parametric representation for the plane of triangle $\mathbf{T}(\mathbf{a},\mathbf{b},\mathbf{c})$ using scalar parameters (r,t) is given by

$$\mathbf{T}(r,t) = \mathbf{c} + r\mathbf{C} - t\mathbf{B} \tag{19}$$

where $\mathbf{B} = \mathbf{c} - \mathbf{b}$ and $\mathbf{C} = \mathbf{a} - \mathbf{c}$. Similarly, the parametric representation of the line segment $\overline{\mathbf{de}}$ is given by

$$\mathbf{l}(s) = \mathbf{d} + s\mathbf{D} \tag{20}$$

The pierce point location $\mathbf{P}(s^*)$ is defined by the intersection of the line and plane. Solving the parametric equations for the point of intersection yields a value of s^* given by

$$\begin{aligned} s^* = &\frac{1}{\Gamma}\{(c_0 - d_0)[(a_2 - c_2)(c_1 - b_1) - (a_1 - c_1)(c_2 - b_2)] \\ &- (c_1 - d_1)[(a_2 - c_2)(c_0 - b_0) - (a_0 - c_0)(c_2 - b_2)] \\ &+ (c_2 - d_2)[(a_1 - c_1)(c_0 - b_0) - (a_0 - c_0)(c_1 - b_1)]\} \end{aligned} \tag{21}$$

where

$$\begin{aligned} \Gamma = &\{(e_0 - d_0)[(a_2 - c_2)(c_1 - b_1) - (a_1 - c_1)(c_2 - b_2)] \\ &- (e_1 - d_1)[(a_2 - c_2)(c_0 - b_0) - (a_0 - c_0)(c_2 - b_2)] \\ &+ (e_2 - d_2)[(a_1 - c_1)(c_0 - b_0) - (a_0 - c_0)(c_1 - b_1)]\} \end{aligned} \tag{22}$$

Similar terms are obtained for r and t.

References

[1] Young, D. P., Melvin, R. G., Bieterman, M. B., Johnson, F. T., and Samant, S. S., "A Locally Refined Rectangular Grid Finite Element Method: Application to Computational Fluid Dynamics and Computational Physics," *Journal of Computational Physics*, Vol. 92, No. 1, 1991, pp. 1–66.

[2] Melton, J. E., *Automated Three-Dimensional Cartesian Grid Generation and Euler Flow Solutions for Arbitrary Geometries*, Ph.D. thesis, University of California Davis, 1996.

[3] Karman Jr., S. L., "SPLITFLOW: A 3D Unstructured Cartesian/Prismatic Grid CFD Code for Complex Geometries," AIAA Paper 95–0343, Jan. 1995.

[4] Charlton, E. F. and Powell, K. G., "An Octree Solution to Conservation-laws over Arbitrary Regions (OSCAR)," AIAA Paper 97–0198, Jan. 1997.

[5] Murman, S. M., Aftosmis, M. J., and Nemec, M., "Automated Parameter Studies Using a Cartesian Method," AIAA Paper 2004–5076, Providence, RI, Aug. 2004.

[6] Aftosmis, M. J., "Solution Adaptive Cartesian Grid Methods for Aerodynamic Flows with Complex Geometries," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 1997-02, Brussels, Belgium, March 1997.

[7] Aftosmis, M. J., Berger, M. J., and Melton, J. E., "Robust and Efficient Cartesian Mesh Generation for Component-Based Geometry," *AIAA Journal*, Vol. 36, No. 6, 1998, pp. 952–960.

[8] Samareh, J. A., "Survey of Shape Parametrization Techniques for High-Fidelity Multidisciplinary Shape Optimization," *AIAA Journal*, Vol. 39, No. 5, 2001, pp. 877–883.

[9] Alonso, J. J., Martins, J. R. R. A., Reuther, J. J., Haimes, R., and Crawford, C., "High-Fidelity Aero-Structural Design Using a Parametric CAD-Based Model," AIAA Paper 2003–3429, Orlando, FL, June 2003.

[10] Samareh, J. A., "Aerodynamic Shape Optimization Based on Free-form Deformation," AIAA Paper 2004–4630, Albany, NY, Sept. 2004.

[11] Fudge, D. M., Zingg, D. W., and Haimes, R., "A CAD-Free and a CAD-Based Geometry Control System for Aerodynamic Shape Optimization," AIAA Paper 2005–0451, Reno, NV, Jan. 2005.

[12] Bentamy, A., Guibault, F., and Trépanier, J. Y., "Aerodynamic Optimization of a Realistic Aircraft Wing," AIAA Paper 2005–332, Jan. 2005.

[13] Dawes, W. N., "Building Blocks Towards VR-Based Flow Sculpting," AIAA Paper 2005–1156, Reno, NV, Jan. 2005.

[14] Nemec, M., Aftosmis, M. J., and Pulliam, T. H., "CAD-Based Aerodynamic Design of Complex Configurations Using a Cartesian Method," AIAA Paper 2004–0113, Reno, NV, Jan. 2004.

[15] Haimes, R. and Follen, G., "Computational Analysis PRogramming Interface," *Proceedings of the 6th International Conference on Numerical Grid Generation in Computational Field Simulations*, edited by Cross, Eiseman, Hauser, Soni, and Thompson, University of Greenwich, 1998.

[16] Haimes, R. and Crawford, C., "Unified Geometry Access for Analysis and Design," Tech. rep., 12th International Meshing Roundtable, Santa Fe, NM, Sept. 2003.

[17] Aftosmis, M. J., Berger, M. J., and Adomavicius, G., "A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries," AIAA Paper 2000–0808, Reno, NV, Jan. 2000.

[18] Aftosmis, M. J., Berger, M. J., and Murman, S. M., "Applications of Space-Filling-Curves to Cartesian Methods for CFD," AIAA Paper 2004–1232, Reno, NV, Jan. 2004.

[19] Pironneau, O., *Optimal Shape Design for Elliptic Systems*, Springer-Verlag, New York, USA, 1984.

[20] Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, 1988, pp. 233–260, Also ICASE report 88–64.

[21] Anderson, W. K. and Venkatakrishnan, V., "Aerodynamic Design Optimization on Unstructured Grids with a Continuous Adjoint Formulation," *Computers & Fluids*, Vol. 28, 1999, pp. 443–480.

[22] Jameson, A., Pierce, N. A., and Martinelli, L., "Optimum Aerodynamic Design using the Navier-Stokes Equations," *Theoretical and Computational Fluid Dynamics*, Vol. 10, No. 1, 1998, pp. 213–237.

[23] Reuther, J. J., Jameson, A., Alonso, J. J., Rimlinger, M. J., and Saunders, D., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1," *Journal of Aircraft*, Vol. 36, No. 1, 1999, pp. 51–60.

[24] Kim, C. S., Kim, C., and Rho, O. H., "Sensitivity Analysis for the Navier–Stokes Equations with Two-Equation Turbulence Models," *AIAA Journal*, Vol. 39, No. 5, 2001, pp. 838–845.

[25] Nemec, M. and Zingg, D. W., "Newton–Krylov Algorithm for Aerodynamic Design Using the Navier–Stokes Equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.

[26] Giles, M. B., Duta, M. C., Müller, J.-D., and Pierce, N. A., "Algorithm Developments for Discrete Adjoint Methods," *AIAA Journal*, Vol. 41, No. 2, 2003, pp. 198–204.

[27] Mavriplis, D. J., "Formulation and Multigrid Solution of the Discrete Adjoint for Optimization Problems on Unstructured Meshes," AIAA Paper 2005–0319, Reno, NV, Jan. 2005.

[28] Nielsen, E. J. and Park, M. A., "Using An Adjoint Approach to Eliminate Mesh Sensitivities in Computational Design," AIAA Paper 2005–0491, Reno, NV, Jan. 2005.

[29] Burdyshaw, C. E. and Anderson, W. K., "A General and Extensible Unstructured Mesh Adjoint Method," AIAA Paper 2005–0335, Reno, NV, Jan. 2005.

[30] Melvin, R. G., Huffman, W. P., Young, D. P., Johnson, F. T., Hilmes, C. L., and Bieterman, M. B., "Recent Progress in Aerodynamic Design Optimization," *International Journal for Numerical Methods in Fluids*, Vol. 30, 1999, pp. 205–216.

[31] Dadone, A. and Grossman, B., "Efficient Fluid Dynamic Design Optimization Using Cartesian Grids," AIAA Paper 2003–3959, Orlando, FL, June 2003.

[32] Dadone, A. and Grossman, B., "Ghost-Cell Method for Inviscid Two-Dimensional Flows on Cartesian Grids," *AIAA Journal*, Vol. 42, No. 12, 2004, pp. 2499–2507.

[33] Nemec, M., Aftosmis, M. J., Murman, S. M., and Pulliam, T. H., "Adjoint Formulation for an Embedded-Boundary Cartesian Method," AIAA Paper 2005–0877, Reno, NV, Jan. 2005.

[34] van Leer, B., "Flux-Vector Splitting for the Euler Equations," ICASE Report 82-30, Sept. 1982.

[35] Berger, M. J., Aftosmis, M. J., and Murman, S. M., "Analysis of Slope Limiters on Irregular Grids," AIAA Paper 2005–0490, Reno, NV, Jan. 2005.

[36] Berger, M. J., Aftosmis, M. J., Marshall, D. D., and Murman, S. M., “Performance of a new CFD flow solver using a hybrid programming paradigm,” *J. Parallel Distrib. Comput.*, Vol. 65, 2005, pp. 414–423.

[37] Giles, M. B., “On the Use of Runge–Kutta Time-Marching and Multigrid for the Solution of Steady Adjoint equations,” Oxford University Computing Laboratory 00/10, June 2000.

[38] Haimes, R. and Aftosmis, M. J., “On Generating High Quality ”Water-tight” Triangulations Directly from CAD,” Tech. rep., Meeting of the International Society for Grid Generation, (ISGG), Honolulu, HI, June 2002.

[39] Haimes, R. and Aftosmis, M. J., “Watertight Anisotropic Surface Meshing Using Quadrilateral Patches,” 13th International Meshing Roundtable, Sept. 2004.

[40] Hicks, R. M. and Henne, P. A., “Wing Design by Numerical Optimization,” *Journal of Aircraft*, Vol. 15, No. 7, 1978, pp. 407–412.

[41] Nielsen, E. J. and Anderson, W. K., “Aerodynamic Design Optimization on Unstructured Meshes Using the Navier Stokes Equations,” *AIAA Journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[42] Jameson, A. and Kim, S., “Reduction of the Adjoint Gradient Formula in the Continuous Limit,” AIAA Paper 2003–0040, Jan. 2003.

[43] Soto, O. and Löhner, R., “On the Computation of Flow Sensitivities from Boundary Integrals,” AIAA Paper 2004–0112, Jan. 2004.

[44] Huffman, W. P., Melvin, R. G., Young, D. P., Johnson, F. T., Bussoletti, J. E., Bieterman, M. B., and Hilmes, C. L., “Practical Design and Optimization in Computational Fluid Dynamics,” AIAA Paper 93–3111, 1993.